École Supérieure Privée d'Ingénierie et de Technologie (TEK-UP)

Rapport de Projet Universitaire

Développement d'une plateforme SaaS de gestion médicale

MedFlow

Encadré par : Mohamed Amine Ben Rhouma

Réalisé par :

Abderahmen Ziedi

Wael Ghribi

Mohamed Amine Kchaw

Matière : Développement Web — React.js

Spécialité : Génie Logiciel et Systèmes d'Information

Année Universitaire 2024–2025

Table des matières

Introduction Générale

La digitalisation des services de santé est devenue un enjeu majeur dans l'amélioration de la qualité des soins, de la fluidité des échanges et de l'accès aux services médicaux. Face à un besoin croissant de modernisation, les professionnels de la santé ainsi que les patients recherchent des solutions fiables, simples d'utilisation et capables d'optimiser la gestion des consultations. C'est dans ce contexte que s'inscrit le développement de **MedFlow**, une application web dédiée à la gestion complète du parcours médical en ligne.

Ce projet vise à offrir une plateforme centralisée permettant aux médecins de gérer efficacement leurs profils, leurs disponibilités, leurs tarifs ainsi que les retours de leurs patients, tout en facilitant aux patients la prise de rendez-vous, le suivi des consultations et le paiement sécurisé des services. Grâce à une architecture moderne reposant sur **Node.js** pour le backend, **React** pour le frontend et **MongoDB** pour la base de données, MedFlow garantit performance, flexibilité et évolutivité. L'intégration de **Stripe** assure par ailleurs une gestion fiable et sécurisée des transactions.

Ce rapport présente les différentes phases de réalisation du projet, depuis la conception et la modélisation des cas d'utilisation jusqu'au développement progressif des fonctionnalités réparties en plusieurs releases. Il décrit également les choix techniques adoptés ainsi que le processus de déploiement en environnement Docker.

Ainsi, MedFlow constitue une contribution concrète à la modernisation du secteur médical en Tunisie, en proposant une solution innovante, ergonomique et adaptée aux attentes des utilisateurs finaux.

Chapitre 1

Cadre général du projet

1.1 Introduction

Dans ce chapitre, nous présentons le contexte général de notre projet universitaire intitulé **MedFlow**. Nous décrivons les motivations ayant conduit à son développement, la problématique identifiée dans le domaine médical, ainsi que les objectifs fixés. Nous exposons également l'étude de l'existant et la méthodologie de travail adoptée pour la conception et la réalisation de l'application web.

1.2 Contexte général

Le secteur de la santé connaît depuis plusieurs années une transformation numérique profonde. Malgré cette évolution, plusieurs difficultés persistent : manque d'outils unifiés pour la gestion des dossiers médicaux, coordination limitée entre les différents acteurs, difficultés d'accès aux soins et à la prise de rendez-vous, et absence de solutions de consultation à distance fiables.

Ces limitations entraînent divers problèmes, tels que :

— une fragmentation des informations médicales,

— une perte de temps liée à la gestion manuelle ou papier des dossiers,

— des erreurs dues à un manque de centralisation,

— une communication insuffisante entre patients et professionnels,

— une expérience utilisateur peu satisfaisante.

Afin de répondre à ces enjeux, notre projet vise à proposer une application web moderne, sécurisée et accessible permettant de centraliser les informations médicales et de fluidifier les interactions entre les utilisateurs.

1.3 Présentation du projet

1.3.1 Cadre général

Le projet **MedFlow** s'inscrit dans un contexte où l'amélioration de l'accès aux soins et la digitalisation des processus médicaux sont devenues des priorités. Les outils actuels restent souvent incomplets, peu ergonomiques ou fragmentés, ce qui renforce le besoin d'une solution intégrée.

MedFlow se présente comme une application web permettant une gestion complète et centralisée des consultations médicales, accessibles depuis n'importe quel navigateur sans installation préalable. Elle est conçue pour simplifier la communication entre patients et médecins, automatiser certains processus et offrir une interface moderne répondant aux standards actuels.

1.3.2 Description du projet

L'objectif principal du projet est de développer une plateforme web offrant un ensemble de fonctionnalités facilitant le suivi et le traitement des consultations médicales. Les principales réalisations envisagées sont les suivantes :

- **Gestion des rendez-vous :** permettre aux patients de réserver, modifier ou annuler des rendez-vous en ligne.
- **Gestion des dossiers médicaux :** centraliser les informations médicales, les antécédents et l'historique des consultations.
- **Messagerie sécurisée :** offrir un espace d'échange entre médecins et patients.
- **Consultations virtuelles :** intégrer un système de visio-consultation sécurisé accessible depuis un navigateur web.
- **Tableau de bord médecin :** permettre aux professionnels de santé de gérer efficacement leurs rendez-vous, consultations et patients.

Ce projet nous permet d'aborder des problématiques techniques réelles tout en développant des compétences en ingénierie logicielle et en développement d'applications web modernes.

1.3.3 Étude de l'existant

Avant la conception de MedFlow, une analyse de solutions existantes a été réalisée. Elle a permis de mettre en évidence plusieurs limites :

- Les plateformes actuelles sont souvent fragmentées : certaines gèrent les rendez-vous mais pas les dossiers médicaux, et inversement.
- L'accès aux consultations à distance n'est pas toujours fluide ou sécurisé.
- Plusieurs outils ne sont pas adaptés à tous les types d'appareils.
- L'expérience utilisateur est parfois peu ergonomique.
- Certaines solutions manquent de souplesse ou sont payantes sans offrir toutes les fonctionnalités nécessaires.

Ces constats justifient le développement d'une solution plus complète, intégrée et centrée sur les besoins des utilisateurs.

1.3.4 Problématique

La problématique principale peut être formulée ainsi :

Comment concevoir une application web moderne, sécurisée et intuitive permettant de centraliser la gestion des consultations médicales et d'améliorer la communication entre patients et professionnels de santé ?

Cette problématique reflète les besoins de performance, de sécurité, d'accessibilité et d'ergonomie indispensables dans une application dédiée au domaine médical.

1.3.5 Solution proposée

Pour répondre à cette problématique, MedFlow propose une solution web intégrée offrant :

— une gestion automatisée des rendez-vous,

— une centralisation complète des dossiers médicaux,

— des consultations virtuelles accessibles via navigateur,

— une interface moderne et intuitive conçue avec React,

— une API sécurisée utilisant Node.js, Express et JWT,

— une base de données flexible et performante grâce à MongoDB.

L'ensemble vise à créer un écosystème numérique cohérent, sécurisé et adapté aux besoins des utilisateurs.

1.4 Méthodologie de travail

1.4.1 Formalismes de modélisation

Pour modéliser et concevoir le système, nous avons utilisé le formalisme UML, un langage graphique orienté objet permettant de représenter les aspects statiques et dynamiques du système. Les principaux diagrammes utilisés sont :

— **Diagrammes de cas d'utilisation** : pour identifier les interactions entre les acteurs et le système.

— **Diagramme de classes** : pour modéliser la structure logique du système.

— **Diagrammes de séquence** : pour décrire les interactions temporelles entre les composants.

1.4.2 Méthodologie adoptée

Le projet suit la méthodologie agile **Scrum**, adaptée au développement de solutions évolutives. Cette méthodologie repose sur un fonctionnement itératif et incrémental permettant :

— une meilleure visibilité sur l'avancement du projet,

— une adaptation continue en fonction des besoins,

— une division du projet en sprints gérables,

— une amélioration progressive grâce aux revues et rétrospectives.

Chaque sprint comprend : la planification, le développement, les tests, la revue et la préparation du sprint suivant.

1.5 Conclusion

Dans ce chapitre, nous avons présenté le cadre général du projet MedFlow, incluant son contexte, sa problématique, l'étude de l'existant et la solution proposée. Nous avons également exposé les formalismes de modélisation et la méthodologie de développement adoptée. Le chapitre suivant sera consacré à l'analyse détaillée des besoins fonctionnels et non fonctionnels du système.

Chapitre 2

Analyse et spécification des besoins

2.1 Introduction

Dans ce chapitre, nous présentons la première phase essentielle de la méthodologie Scrum, souvent appelée *sprint zéro* ou phase de "planification et architecture". Cette étape vise à établir une vision globale du produit, identifier les acteurs qui interagiront avec le système MedFlow, définir les besoins fonctionnels et non fonctionnels, élaborer la planification des releases et structurer les sprints du projet.

L'objectif principal est de traduire les attentes des utilisateurs (patients et médecins) en exigences formelles permettant d'orienter la conception et le développement de la plateforme.

2.2 Spécification des besoins

2.2.1 Identification des acteurs

Le système MedFlow comporte deux acteurs principaux dont les rôles et responsabilités diffèrent :

- **Patient** : consulte son dossier médical, prend des rendez-vous, participe aux consultations virtuelles, effectue des paiements en ligne et suit son historique de santé.
- **Médecin** : gère ses rendez-vous, consulte et met à jour les dossiers médicaux, interagit avec les patients et accède aux statistiques relatives à son activité.

2.2.2 Besoins fonctionnels par acteur

Afin de mieux cerner les interactions entre chaque acteur et le système, les besoins fonctionnels sont décrits ci-dessous selon leur perspective.

Le Patient

- **Gestion de l'accès** :
  - S'authentifier et se déconnecter de manière sécurisée.
- **Gestion des rendez-vous** :
  - Prendre un rendez-vous en ligne.

  - Participer à une consultation virtuelle.
  - Annuler ou modifier un rendez-vous.
- **Gestion du dossier médical** :
  - Consulter son dossier médical.
  - Accéder à l'historique des rendez-vous.
- **Paiement en ligne** :
  - Effectuer un paiement sécurisé via Stripe.
  - Consulter l'historique des paiements.
- **Feedback** :
  - Évaluer le médecin après la consultation.

Le Médecin

- **Gestion de l'accès** :
  - Authentification sécurisée selon le rôle.
- **Gestion des consultations** :
  - Consulter les rendez-vous planifiés.
  - Accepter ou rejeter une demande de rendez-vous.
  - Accéder aux dossiers médicaux des patients.
  - Consulter l'historique des interactions.
- **Statistiques et performance** :
  - Consulter les statistiques d'activité (nombre de consultations, revenus, évaluations).
- **Feedback** :
  - Consulter les évaluations des patients.

2.2.3 Besoins non fonctionnels

Les besoins non fonctionnels définissent les contraintes et exigences qualitatives du système MedFlow :

- **Sécurité** : utilisation de JWT, chiffrement des données, conformité aux bonnes pratiques de sécurité.
- **Ergonomie** : interface intuitive, homogène et adaptée à tous types d'utilisateurs.
- **Performance** : temps de réponse optimisé, gestion efficace des requêtes.
- **Évolutivité** : architecture modulaire facilitant l'ajout ou la modification de fonctionnalités.

— **Compatibilité** : application responsive et compatible avec tout type de support.

2.3 Diagramme de cas d'utilisation global

Le diagramme de cas d'utilisation global présente une vision abstraite des interactions entre les acteurs et les fonctionnalités principales du système.

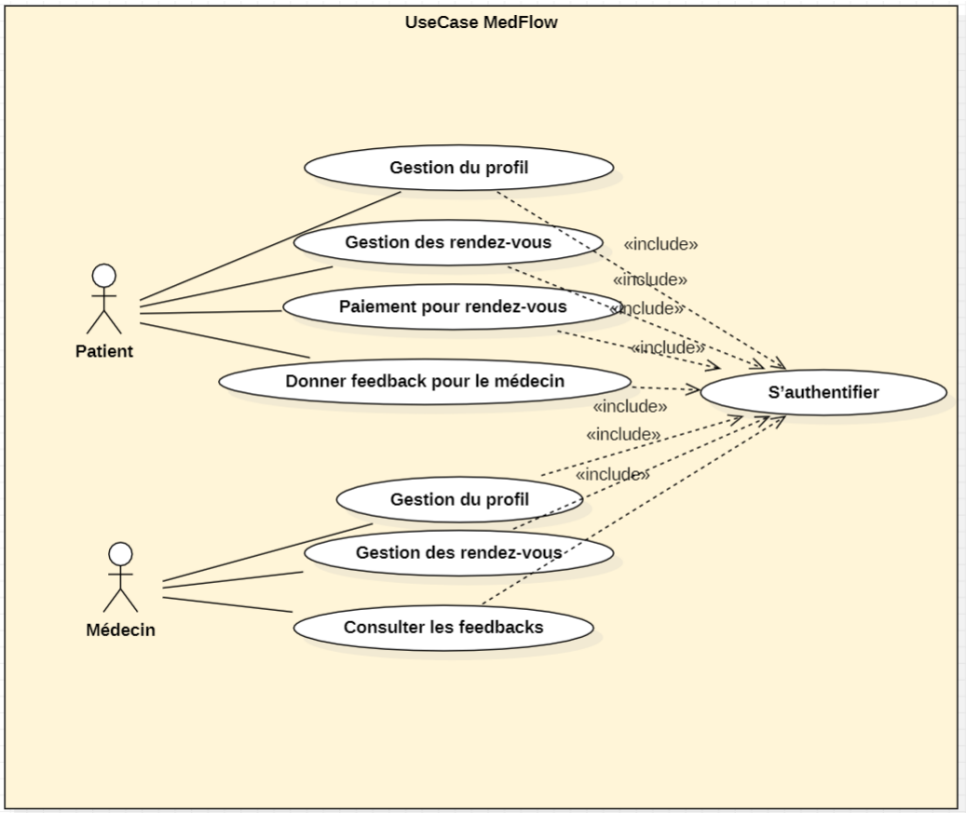

FIGURE 2.1 – Diagramme de cas d'utilisation global de MedFlow

2.4 Diagramme de classes

Le diagramme de classes détaille les entités principales du système et les relations entre elles : patients, médecins, rendez-vous, dossiers médicaux et transactions.

2.5 Structure et découpage du projet

2.5.1 Identification de l'équipe Scrum

Le projet MedFlow est réalisé selon la méthodologie Scrum. L'équipe est composée comme suit :

— **Product Owner** : M. Mohamed Amine Ben Rhouma.

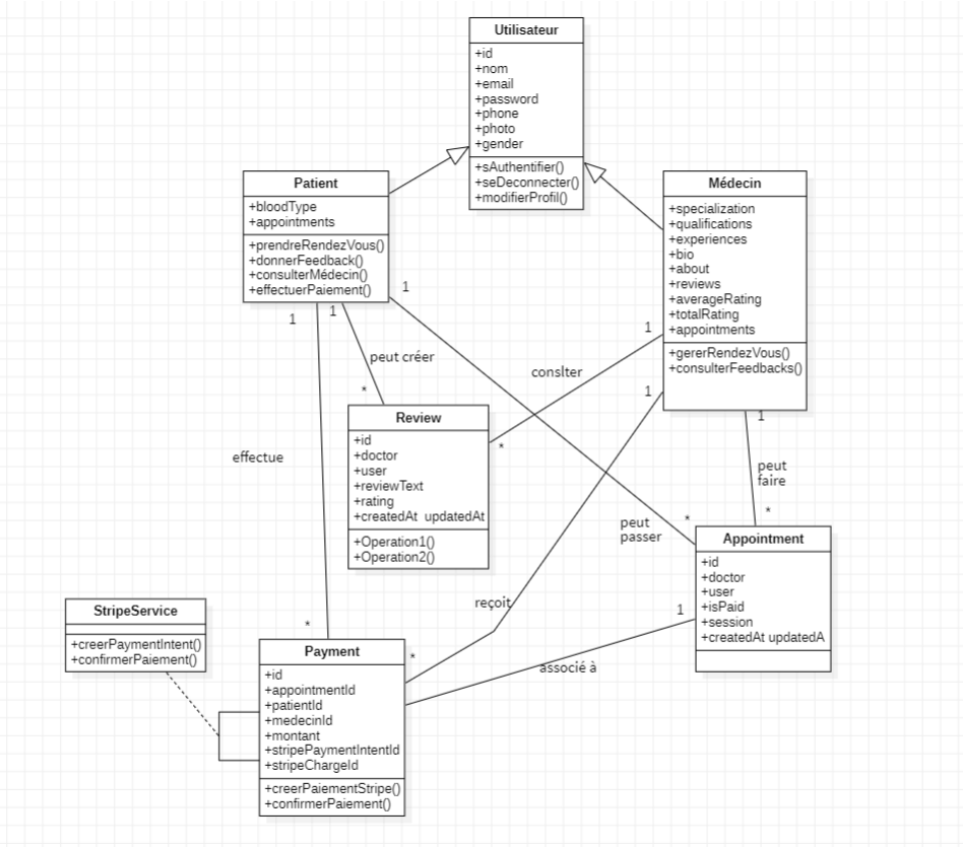

FIGURE 2.2 – Diagramme de classes de MedFlow

— **Équipe de développement** : Abderahmen Ziedi, Wael Ghribi et Mohamed Amine Kchaw.

2.5.2 Backlog produit global

Le backlog du produit regroupe l'ensemble des *User Stories* classées par priorité. Il constitue la base de la planification des releases et sprints.

2.6 Planification des phases du projet

La planification du projet suit une approche incrémentale articulée autour de trois releases principales.

2.7 Planification des phases du projet

La planification du projet suit une approche incrémentale articulée autour de quatre releases principales.

Phase / Release	Durée	Livrables
Release 1 : Authentification et Comptes	2 semaines	Authentification sécurisée, création et gestion des comptes.
Sprint 1 : Authentification	1 semaine	JWT, API sécurisée, interface de connexion.
Sprint 2 : Gestion des rôles et comptes	1 semaine	Création des comptes patients/médecins.
Release 2 : Profils et Dashboards	2 semaines	Profils détaillés, dashboards interactifs.
Sprint 1 : Profil patient	1 semaine	Gestion de profil, feedbacks.
Sprint 2 : Profil médecin	1 semaine	Gestion de profil, feedbacks.
Release 3 : Gestion des rendez-vous	2 semaines	Fonctionnalités de planification et réservation des rendez-vous.
Sprint 1 : Gestion des disponibilités (Médecin)	1 semaine	Le médecin peut définir les dates et créneaux disponibles pour les rendez-vous, ainsi que le prix par consultation.
Sprint 2 : Prise de rendez-vous (Patient)	1 semaine	Le patient peut lister les médecins disponibles, choisir un médecin, sélectionner un créneau proposé et réserver le rendez-vous.
Release 4 : Paiement et Déploiement	2 semaines	Paiement Stripe, gestion des transactions et mise en production.
Sprint 1 : Paiement Stripe	1 semaine	Réservation avec paiement en ligne.
Sprint 2 : Déploiement final	1 semaine	Déploiement Front/Back, configuration du domaine, HTTPS, optimisation production.

TABLE 2.1 – Planification des releases et sprints du projet MedFlow (version révisée)

2.8 Conclusion

Ce chapitre a posé les bases essentielles du projet MedFlow en identifiant les besoins, les acteurs, les diagrammes UML, ainsi que la structure générale du projet. La planification réalisée permettra d'assurer une organisation efficace du développement et une progression cohérente selon les principes Scrum.

Chapitre 3

Release 1 : Authentification et Gestion des Comptes

3.1 Introduction

La Release 1 constitue la base du fonctionnement de l'application **MedFlow** en mettant en place deux éléments fondamentaux :

— l'authentification sécurisée des utilisateurs ;

— la création et la gestion des comptes pour les deux profils : *Patient* et *Médecin*.

Cette version s'appuie sur une architecture moderne utilisant **ReactJS** pour le *frontend*, **Node.js** pour le *backend* et **MongoDB** pour la base de données.

3.2 Sprint 1 : Authentification

3.2.1 Objectifs

— Mettre en place un système d'authentification sécurisé basé sur JWT.

— Restreindre l'accès aux fonctionnalités selon les rôles utilisateurs.

— Garantir la sécurité et la bonne gestion des tokens côté client.

3.2.2 Backlog

— Configuration et intégration de JWT dans le backend Node.js.

— Développement des interfaces de connexion sous ReactJS.

— Mise en place du contrôle d'accès selon les rôles *Patient* et *Médecin*.

— Stockage sécurisé des tokens côté client.

3.2.3 Réalisation

— Intégration de JWT pour la sécurisation des endpoints.

— Développement de la page de connexion et validation des tokens reçus.

— Restriction des accès aux différentes fonctionnalités en fonction des rôles.

3.2.4 Diagramme de séquence : Authentification

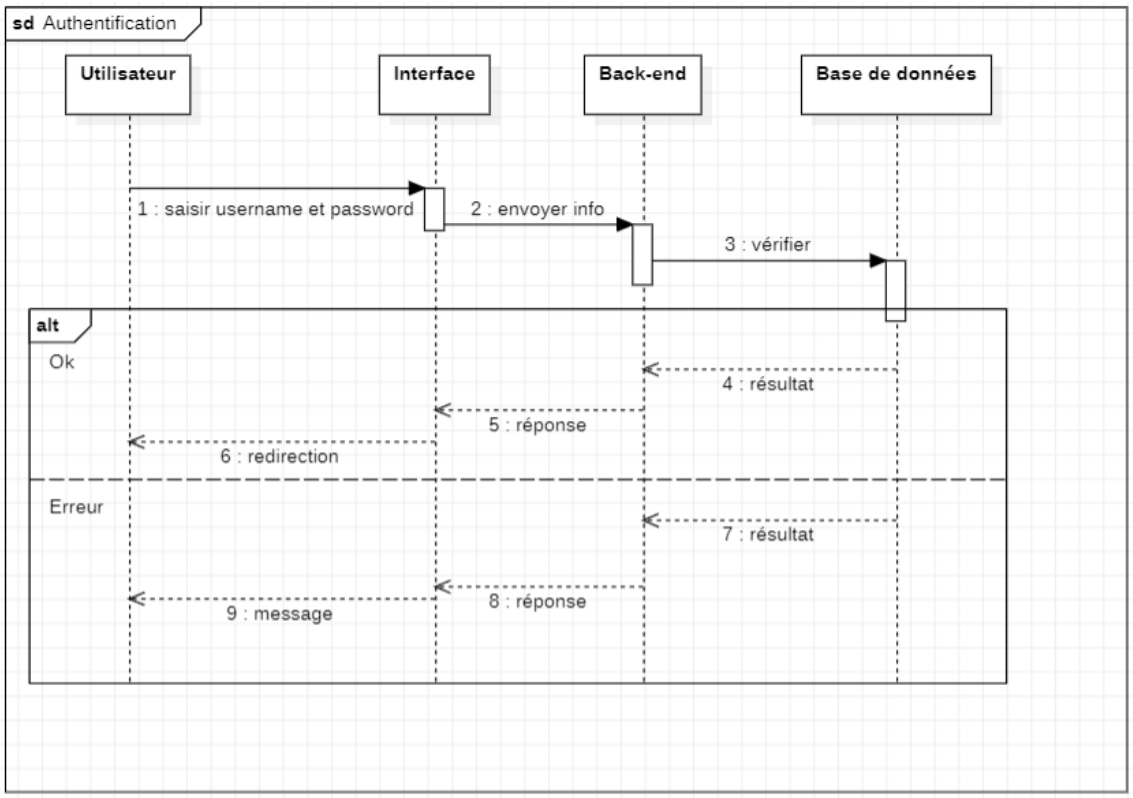

FIGURE 3.1 – Diagramme de séquence de l'authentification dans MedFlow

3.2.5 Conclusion du Sprint 1

Ce sprint a permis de sécuriser l'accès au système et d'établir une séparation claire entre les rôles, assurant ainsi les fondations de l'accès différencié.

3.3 Sprint 2 : Création des comptes et gestion des rôles

3.3.1 Objectifs

— Permettre la création de comptes utilisateurs.

— Associer un rôle (*Patient* ou *Médecin*) lors de l'inscription.

— Garantir la cohérence et la validité des données saisies.

3.3.2 Backlog

— Conception des formulaires d'inscription sous ReactJS.

— Développement des endpoints Node.js dédiés à la création de comptes.

— Validation et enregistrement des informations dans MongoDB.

— Association correcte du rôle choisi lors de l'inscription.

3.3.3 Use Case : Création d'un compte selon le rôle

Nom du cas d'utilisation : Création d'un compte utilisateur
Acteurs : Utilisateur (Patient ou Médecin), Système MedFlow
Objectif : Permettre à un nouvel utilisateur de créer un compte et d'obtenir un rôle déterminé.
Précondition : L'utilisateur n'est pas encore enregistré dans le système.
Postcondition : Un nouveau compte est créé et stocké dans MongoDB avec le rôle adéquat.

Scénario principal

1. L'utilisateur accède à la page d'inscription.
2. Le système affiche un formulaire comprenant : informations personnelles, e-mail, mot de passe, rôle (*Patient* ou *Médecin*).
3. L'utilisateur saisit les informations demandées.
4. L'utilisateur choisit son rôle dans la liste déroulante.
5. L'utilisateur valide le formulaire.
6. Le système vérifie la validité des données (format email, mot de passe, champs obligatoires).
7. Le système vérifie l'unicité de l'e-mail.
8. En cas d'erreur, le système renvoie un message explicatif.
9. Si les données sont valides, le système crée un nouvel utilisateur en associant le rôle sélectionné.
10. Le système enregistre les données dans MongoDB.
11. Le système confirme la création du compte.

Scénarios alternatifs

A1 — Email déjà utilisé

— 6a. Le système détecte que l'e-mail existe déjà.

— 6b. Un message d'erreur est affiché : « Cet e-mail est déjà associé à un compte ».

— 6c. L'utilisateur peut modifier son adresse et réessayer.

A2 — Données non valides

— 6a. Le système détecte des champs invalides (mot de passe faible, format email incorrect).

— 6b. Le système affiche les erreurs correspondantes.

— 6c. L'utilisateur corrige et renvoie les informations.

A3 — Rôle non sélectionné

— 4a. L'utilisateur ne sélectionne pas de rôle.

— 4b. Le système refuse l'envoi et demande la sélection obligatoire d'un rôle.

3.3.4 Réalisation

— Développement d'un formulaire complet d'inscription avec sélection du rôle.

— Validation des données côté client et côté serveur.

— Enregistrement sécurisé des informations dans MongoDB avec hachage du mot de passe.

— Attribution automatique du rôle permettant le contrôle d'accès ultérieur.

3.3.5 Conclusion du Sprint 2

À l'issue de ce sprint, la création et la gestion des comptes sont opérationnelles. Le système distingue correctement les utilisateurs selon leur rôle et garantit une base fiable pour les fonctionnalités futures.

3.4 Conclusion de la Release 1

La Release 1 garantit :

— une authentification sécurisée et adaptée aux deux rôles principaux ;

— une création et gestion des comptes optimisée et fiable, avec attribution automatique du rôle.

Cette base solide permet d'aborder la Release 2, dédiée à la gestion des profils, aux tableaux de bord et au suivi des consultations.

Chapitre 4

Release 2 : Profils et Dashboards

4.1 Sprint 1 : Gestion du Profil et Feedback du Patient

4.1.1 Titre du Cas d'Utilisation

Gestion du profil patient et soumission de feedback.

4.1.2 Acteurs

Principal : Patient **Secondaires :**

— Médecin (récepteur du feedback)

— Système MedFlow (backend et base de données)

4.1.3 Description

Ce cas d'utilisation décrit le processus permettant au patient de gérer son profil personnel et de soumettre une évaluation (note sur 5 étoiles et commentaire) à un médecin après une consultation.

4.1.4 Préconditions

— Le patient est authentifié dans le système.

— Le patient a déjà effectué une consultation avec le médecin à évaluer.

— Les API nécessaires pour la gestion du profil et l'enregistrement des feedbacks sont disponibles.

4.1.5 Déclencheur

Le patient accède à son espace personnel ou à la page d'évaluation d'un médecin.

4.1.6 Scénario Nominal

1. Le patient ouvre son tableau de bord.
2. Le système affiche les informations actuelles du profil (nom, photo, genre, groupe sanguin, etc.).
3. Le patient sélectionne l'option *Modifier Profil*.
4. Le système affiche un formulaire contenant les champs modifiables :

- Photo de profil
- Nom et prénom
- Genre
- Groupe sanguin

5. Le patient modifie les informations souhaitées et valide les changements.
6. Le système met à jour la base de données et renvoie un message de confirmation.
7. Le patient accède à la liste de ses rendez-vous passés et sélectionne un médecin à évaluer.
8. Le système affiche le formulaire de feedback, incluant :
   - Sélection de la note (1 à 5 étoiles)
   - Champ texte pour le commentaire
9. Le patient saisit sa note et son commentaire, puis valide l'envoi.
10. Le système enregistre le feedback et confirme la soumission.
11. Le feedback devient visible pour le médecin concerné.

4.1.7 Scénarios Alternatifs

- **Informations incomplètes :** le système demande au patient de compléter certains champs avant validation.

4.1.8 Postconditions

- Le profil du patient est mis à jour.
- Le feedback est enregistré et associé à la consultation et au médecin évalué.

4.1.9 Diagrammes

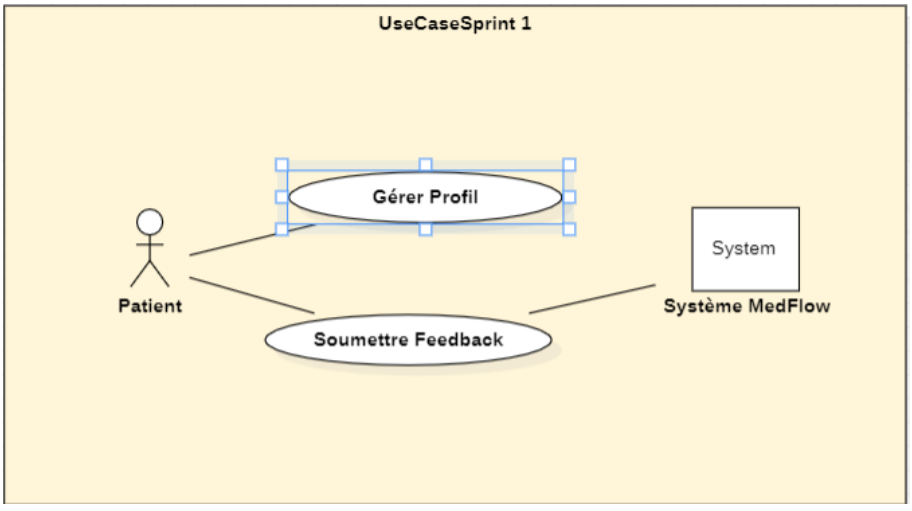

FIGURE 4.1 – Diagramme de cas d'utilisation - Patient

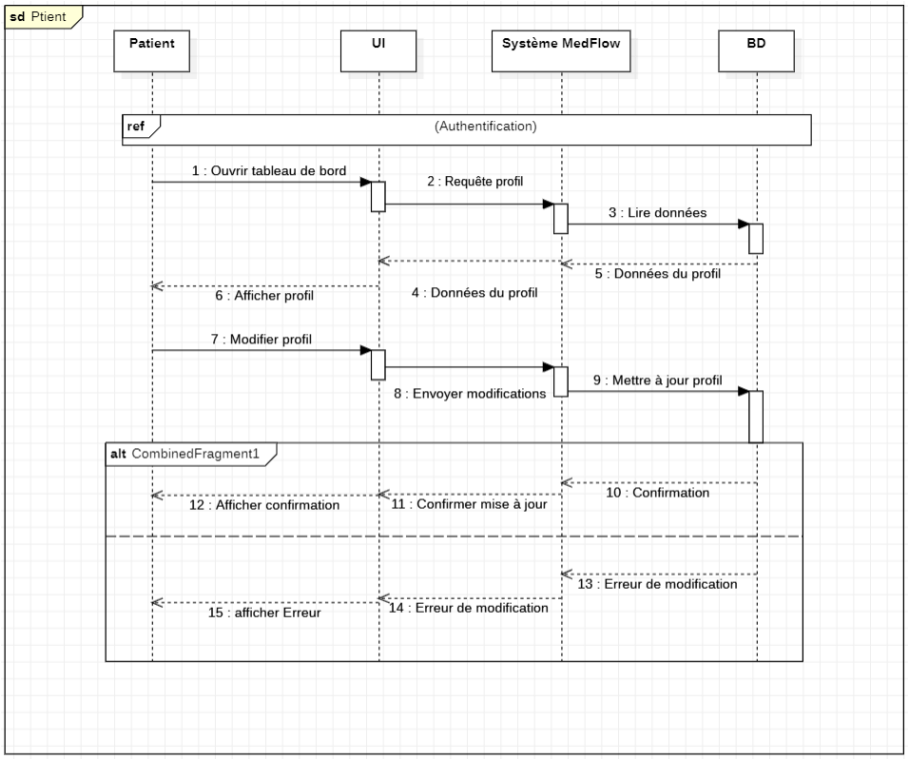

FIGURE 4.2 – Diagramme de séquence - Patient modifiant profil et soumettant feedback

4.2 Sprint 2 : Gestion du Profil et Consultation des Feedbacks du Médecin

4.2.1 Titre du Cas d'Utilisation

Gestion du profil médecin et consultation des feedbacks patients.

4.2.2 Acteurs

Principal : Médecin **Secondaires :**

— Patient (auteur du feedback)

— Système MedFlow

4.2.3 Description

Ce cas d'utilisation décrit la gestion complète du profil professionnel du médecin ainsi que la consultation des notes et commentaires laissés par les patients.

4.2.4 Préconditions

— Le médecin est connecté au système.

— Au moins un patient a laissé un feedback.

— Les API nécessaires pour la mise à jour du profil et la récupération des feedbacks sont disponibles.

4.2.5 Déclencheur

Le médecin accède à son tableau de bord ou à son espace de gestion de profil.

4.2.6 Scénario Nominal

1. Le médecin accède à son espace personnel.
2. Le système affiche les informations actuelles du profil :
   — Photo de profil
   — Genre
   — Biographie
   — Spécialisation
   — Qualifications
   — Expériences professionnelles
   — Créneaux horaires disponibles
   — Prix par consultation
3. Le médecin sélectionne l'option *Modifier Profil*.
4. Le système affiche un formulaire permettant de modifier les informations personnelles et professionnelles.
5. Le médecin met à jour les informations souhaitées et valide les changements.
6. Le système met à jour la base de données et confirme la modification.
7. Le médecin accède à la section *Feedbacks*.
8. Le système affiche tous les feedbacks reçus, incluant :
   — Note (1 à 5 étoiles)
   — Commentaire
   — Nom du patient
   — Date de la consultation évaluée
9. Le médecin consulte et analyse la satisfaction des patients.
10. Le médecin peut filtrer les feedbacks par note, date ou patient.

4.2.7 Scénarios Alternatifs

— **Informations incomplètes :** le système demande au médecin de compléter certains champs avant validation.

— **Aucun feedback disponible :** le système affiche « Aucun feedback pour le moment ».

4.2.8 Postconditions

— Le profil du médecin est mis à jour.

— Tous les feedbacks des patients sont disponibles pour consultation et analyse.

4.2.9 Diagrammes

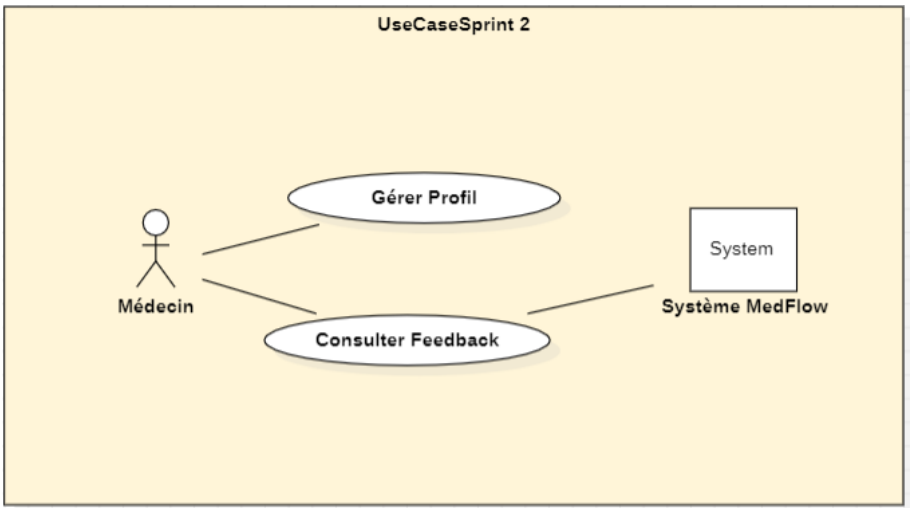

FIGURE 4.3 – Diagramme de cas d'utilisation - Médecin

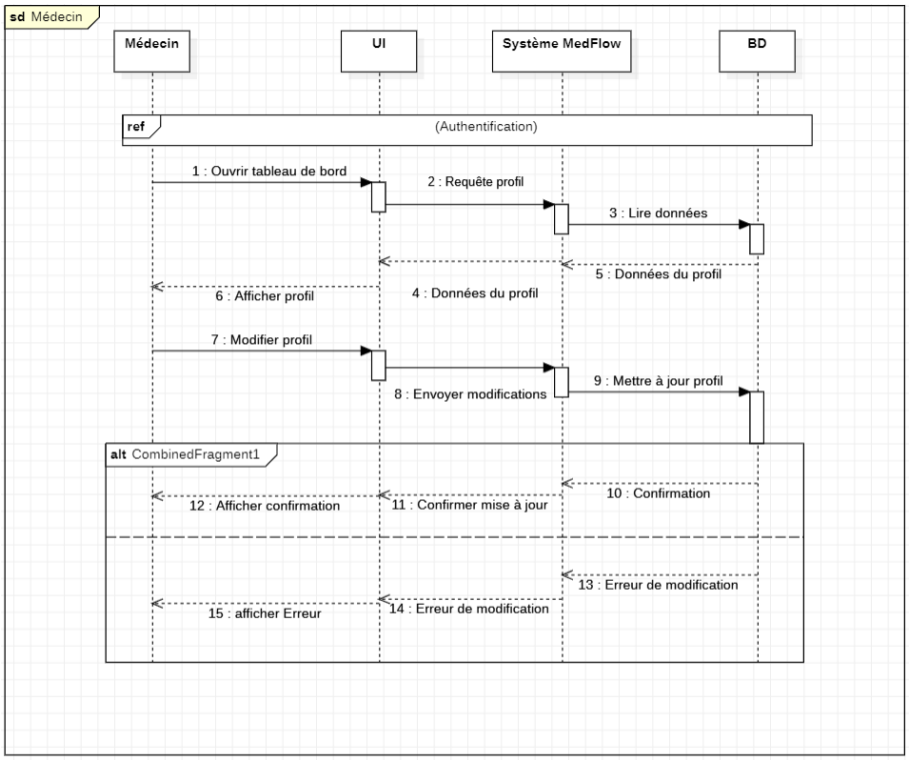

FIGURE 4.4 – Diagramme de séquence - Médecin modifiant profil et consultant feedbacks

Chapitre 5

Release 3 : Gestion des rendez-vous

5.1 Sprint 1 : Gestion des rendez-vous (Médecin)

5.1.1 Titre

Définition des disponibilités et tarifs par le Médecin

5.1.2 Acteur principal

Médecin

5.1.3 Acteurs secondaires

— Système MedFlow (backend)

— Base de données

5.1.4 Description

Ce cas d'utilisation décrit le processus par lequel un médecin définit ses dates, créneaux horaires disponibles et le prix par consultation pour permettre aux patients de prendre rendez-vous.

5.1.5 Préconditions

— Le médecin est authentifié dans le système.

— Le backend est opérationnel et la base de données est accessible.

5.1.6 Déclencheur

Le médecin décide de mettre à jour ou ajouter ses disponibilités pour les rendez-vous.

5.1.7 Scénario nominal

1. Le médecin accède à la section "Profil".
2. Il sélectionne les dates et créneaux horaires disponibles.
3. Il saisit le prix par consultation pour chaque créneau.
4. Le système enregistre les informations dans la base de données.
5. Le médecin reçoit une confirmation que ses disponibilités ont été mises à jour.

5.1.8 Scénarios alternatifs

— **Erreur d'ajout :** si le prix ou la date (jour de la semaine) du créneau n'est pas identifié, l'ajout est obligatoire et le système affiche un message d'erreur demandant de renseigner ces informations.

— **Erreur serveur :** en cas de problème technique, le médecin peut réessayer plus tard.

5.1.9 Postconditions

— Les disponibilités du médecin sont correctement enregistrées.

— Les patients peuvent voir et réserver ces créneaux.

5.1.10 Diagramme de séquence

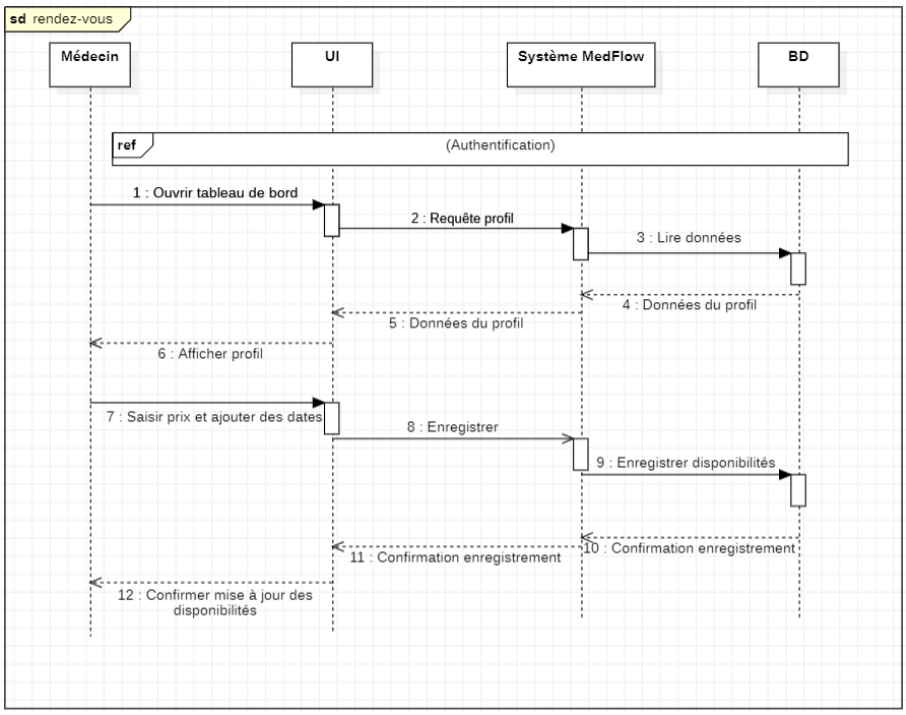

FIGURE 5.1 – Diagramme de séquence : Création des disponibilités par le médecin

5.2 Sprint 2 : Prise de rendez-vous (Patient)

5.2.1 Titre

Réservation d'un rendez-vous par le Patient

5.2.2 Acteur principal

Patient

5.2.3 Acteurs secondaires

— Médecin (fournisseur du créneau)

— Système MedFlow (backend)

— Base de données

5.2.4 Description

Ce cas d'utilisation décrit comment un patient peut consulter les médecins disponibles, choisir un créneau et réserver un rendez-vous.

5.2.5 Préconditions

— Le patient est authentifié dans le système.

— Les disponibilités des médecins sont définies et accessibles.

5.2.6 Déclencheur

Le patient souhaite réserver un rendez-vous avec un médecin.

5.2.7 Scénario nominal

1. Le patient accède à la section "les Docteurs".
2. Il liste les médecins disponibles et sélectionne un médecin.
3. Il consulte les créneaux horaires proposés et choisit un créneau.
4. Le système enregistre le rendez-vous dans la base de données.
5. Le patient et le médecin reçoivent une notif ication de confirmation du rendez-vous.

5.2.8 Scénarios alternatifs

— **Erreur serveur :** en cas de problème technique, le patient peut réessayer ultérieurement.

5.2.9 Postconditions

— Le rendez-vous est enregistré avec succès.

— Le médecin et le patient sont notifiés du rendez-vous confirmé.

5.2.10 Diagramme de séquence

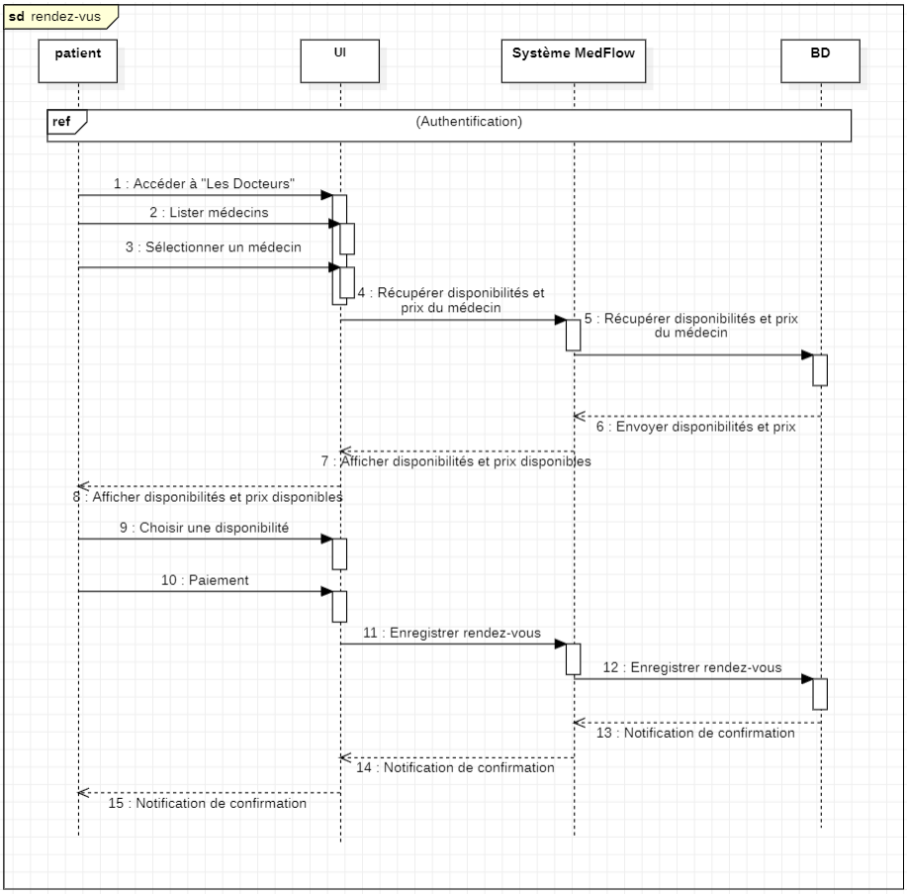

FIGURE 5.2 – Diagramme de séquence : Prise de rendez-vous par le patient

Chapitre 6

Release 3 : Paiement, Déploiement Docker et Kubernetes

6.1 Sprint 1 : Patient - Paiement avec Stripe

6.1.1 Titre

Paiement d'une Consultation par le Patient

6.1.2 Acteur principal

Patient

6.1.3 Acteurs secondaires

— Médecin (bénéficiaire du paiement)

— Système MedFlow (backend)

— Stripe (plateforme de paiement)

6.1.4 Description

Ce cas d'utilisation décrit le processus par lequel un patient effectue le paiement d'une consultation en utilisant Stripe.

6.1.5 Préconditions

— Le patient est authentifié dans le système.

— Une consultation est programmée avec un médecin.

— Le backend est configuré avec l'API Stripe.

6.1.6 Déclencheur

Le patient choisit de payer une consultation depuis son tableau de bord.

6.1.7 Scénario nominal

1. Le patient accède à la page de paiement.
2. Le système récupère le montant de la consultation depuis la base de données.
3. Le patient saisit ses informations de paiement ou utilise un mode de paiement enregistré.

4. Le système envoie les informations à Stripe pour traitement.
5. Stripe autorise et confirme le paiement.
6. Le backend enregistre la transaction comme réussie.
7. Le patient reçoit une confirmation du paiement et un reçu.
8. Le médecin reçoit une notification de paiement confirmé.

6.1.8 Scénarios alternatifs

— **Paiement refusé :** Stripe rejette la transaction. Le système affiche un message d'erreur et propose un autre mode de paiement.

— **Problème réseau :** Si la transaction échoue pour des raisons techniques, le patient peut réessayer.

6.1.9 Postconditions

— Le paiement est enregistré dans la base de données.

— Le patient et le médecin sont notifiés du statut de la transaction.

6.2 Sprint 2 : Déploiement de l'application avec Docker

6.2.1 Titre

Déploiement de MedFlow en Environnement Docker

6.2.2 Acteur principal

Administrateur / DevOps

6.2.3 Acteurs secondaires

— Serveur de production

— Docker Engine

— Base de données MongoDB (conteneur)

6.2.4 Description

Ce cas d'utilisation décrit le processus de déploiement complet de l'application MedFlow à l'aide de Docker et des conteneurs.

6.2.5 Préconditions

— Le code de l'application est disponible dans le dépôt Git.

— Docker et Docker Compose sont installés sur le serveur.

— Les fichiers `Dockerfile` et `docker-compose.yml` sont configurés.

6.2.6 Déclencheur

L'administrateur décide de déployer ou de mettre à jour MedFlow sur le serveur de production.

6.2.7 Scénario nominal

1. L'administrateur clone le dépôt Git ou met à jour la version locale.
2. Il construit les images Docker pour le backend et le frontend :
   - `docker build -t medflow-backend ./backend`
   - `docker build -t medflow-frontend ./frontend`
3. Il lance Docker Compose pour démarrer l'ensemble des conteneurs (backend, frontend, MongoDB).
4. Docker initialise les conteneurs et leurs services.
5. L'administrateur vérifie les logs pour s'assurer du bon démarrage.
6. L'application devient accessible via l'URL publique.

6.2.8 Scénarios alternatifs

— **Échec du build :** l'image ne se construit pas, un message d'erreur apparaît.

— **Port occupé :** l'administrateur modifie la configuration du port dans `docker-compose.yml`.

6.2.9 Postconditions

— L'application MedFlow est déployée et fonctionnelle.

— Les services sont orchestrés via Docker Compose.

6.3 Sprint 3 : Déploiement avancé avec Kubernetes

6.3.1 Titre

Orchestration et Déploiement de MedFlow avec Kubernetes

6.3.2 Acteur principal

Administrateur / DevOps

6.3.3 Acteurs secondaires

— Cluster Kubernetes

— Docker Registry (Docker Hub ou privé)

— Ingress Controller

— MongoDB (StatefulSet ou service externe)

6.3.4 Description

Ce cas d'utilisation décrit le déploiement de l'application MedFlow dans un environnement Kubernetes afin d'assurer une haute disponibilité, une scalabilité automatique et une gestion avancée des microservices.

6.3.5 Préconditions

— Le cluster Kubernetes est opérationnel (Minikube, K3s ou Cloud).

— Les images Docker sont publiées sur un registre accessible.

— Les fichiers YAML (Deployment, Service, Ingress, ConfigMap, Secret) sont configurés.

6.3.6 Déclencheur

Le DevOps décide de déployer MedFlow en production sur Kubernetes.

6.3.7 Scénario nominal

1. Le DevOps publie les images Docker du backend et frontend sur un registre.
2. Il applique les fichiers de déploiement Kubernetes :
   — `kubectl apply -f backend-deployment.yml`
   — `kubectl apply -f frontend-deployment.yml`
   — `kubectl apply -f mongodb-statefulset.yml`
   — `kubectl apply -f ingress.yml`
3. Kubernetes crée les Pods via les Deployments.
4. Les services internes sont exposés via les Services (ClusterIP / NodePort).
5. L'Ingress Controller expose l'application sur un domaine public.
6. Le DevOps vérifie l'état :
   — `kubectl get pods`
   — `kubectl get svc`
   — `kubectl get ingress`
7. Kubernetes assure automatiquement la scalabilité et le redémarrage des Pods en cas d'échec.

6.3.8 Scénarios alternatifs

— **Image introuvable :** le registre ne contient pas l'image demandée.

— **Pods en CrashLoopBackOff :** problème dans l'image Docker ou les variables d'environnement.

— **Ingress non accessible :** problème de DNS ou de configuration Nginx.

6.3.9 Postconditions

— L'application MedFlow est déployée et offerte en haute disponibilité.

— Kubernetes garantit auto-scalabilité, reprise automatique et supervision continue.

6.4 Schéma explicatif du déploiement Kubernetes

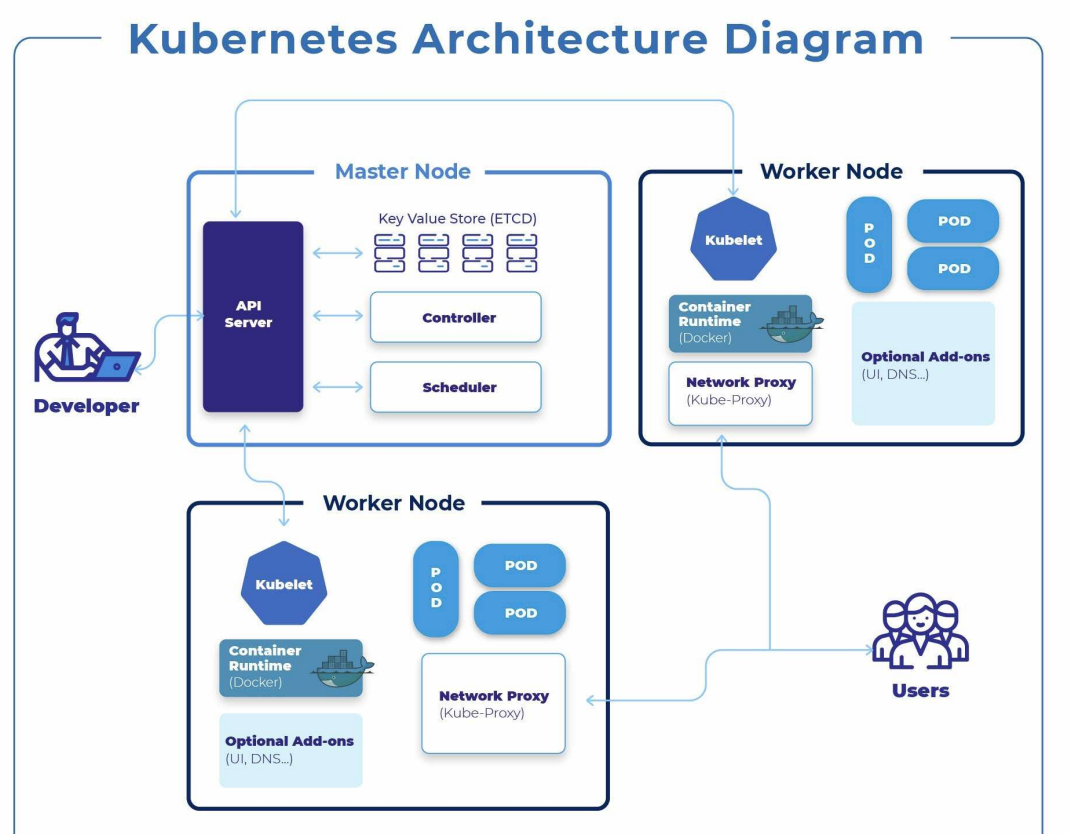

FIGURE 6.1 – Architecture de déploiement de MedFlow sur Kubernetes

Conclusion

Le projet **MedFlow** a permis de concevoir et de développer une application web moderne dédiée à la gestion des interactions entre médecins et patients. Cette solution améliore significativement l'organisation des consultations en intégrant des fonctionnalités essentielles telles que la gestion des profils, la définition des disponibilités, la prise de rendez-vous et le suivi des paiements en ligne. Elle contribue ainsi à la digitalisation du parcours médical et à l'optimisation du travail des professionnels de santé.

L'utilisation de technologies actuelles telles que **Node.js**, **React** et **MongoDB**, combinée à l'intégration du prestataire de paiement **Stripe**, a permis de garantir une architecture performante, sécurisée et évolutive. De plus, le déploiement via **Docker** a facilité l'orchestration des services et assuré une mise en production fiable. Ce projet a également été l'occasion d'approfondir mes compétences en développement web, en modélisation UML et en conception d'applications distribuées.

En somme, MedFlow constitue une solution innovante contribuant à la modernisation du secteur médical en Tunisie, en offrant un outil efficace, sécurisé et parfaitement adapté aux besoins des médecins et des patients. Le système posé ouvre également la voie à de futures améliorations telles que la téléconsultation, la gestion des dossiers médicaux électroniques ou l'analyse intelligente des données de santé.